3. Валькман Ю.Р., Быков В.С. О моделировании образного мышления в компьютерных технологиях: общие закономерности мышления. // Сборник научных трудов KDS-2005 20-30 июня 2005 г., т.1., София, 2005, с.37-45.
4. Гладун В.П., Величко В.Ю. Конспектирование естественно-языковых текстов. // Сборник научных трудов KDS-2005 20-30 июня 2005 г., т.2., София, 2005, с.344-347.
5. Леонтьева Н.Н., Семенова С.Ю. Семантический словарь РУСЛАН как инструмент компьютерного понимания. // Понимание в коммуникации. Материалы научно-практической конф. 5-6 марта 2003 г. — М., МГГИИ, 2003. – С.41-46.
6. Жермен-Ликур П., Жорж П., Пистр Ф., Безье П. Математика и САПР: в 2-х кн. Кн.2-М.: Мир, 1988.- 264с.
7. Люгер Дж.Ф. Искусственный интеллект: Стратегии и методы решения сложных проблем. - М.: Изд. дом "Вильямс". 2003.- 864с.
8. Эгрон Ж. Синтез изображений : Базовые алгоритмы. -М.: Радио и связь, 1993.-216с.
9. Роджерс Д. Алгоритмические основы машинной графики. - М.: Мир, 1989. -512с.

Author's Information

Nikolay Borysovitch Fesenko – V.M.Glushkov Institute of cybernetics NAS of Ukraine, Kiev-187 ГСП, 03680, Akademik Glushov avenue, 40,e-mail: glad@aduis.kiev.ua

AUTOMATED PROBLEM DOMAIN COGNITION PROCESS IN INFORMATION SYSTEMS DESIGN

Maxim Loginov, Alexander Mikov

Abstract: An automated cognitive approach for the design of Information Systems is presented. It is supposed to be used at the very beginning of the design process, between the stages of requirements determination and analysis, including the stage of analysis. In the context of the approach used either UML or ERD notations may be used for model representation. The approach provides the opportunity of using natural language text documents as a source of knowledge for automated problem domain model generation. It also simplifies the process of modelling by assisting the human user during the whole period of working upon the model (using UML or ERD notations).

Keywords: intellectual modeling technologies, information system development, structural analysis of loosely structured natural language documents.

ACM Classification Keywords: I.2 Artificial Intelligence: I.2.7 Natural Language Processing – Text analysis

Introduction

The term "Problem domain" is usually used when the problem of Information Systems (IS) design is discussed. This term represents the aggregation of knowledge about objects and active subjects, tied together with specific relations and pertaining to some common tasks.

Usually the scope of the problem domain is not described strictly and different problem domains intersect. Let us take two problem domains for example: a school education service and a public health service.

An information system designed for automating reporting at schools and another one designed for decision-making for health authorities of a city council can not be completely independent. There are medical consulting rooms at schools and the rate of sickness certainly depends on the school working conditions and so on. After all,

both information systems share some personality information: many people are citizens and students at the same time.

Nevertheless, a description (a model) of the problem domain is a very important part of an information system project. But, anyway, if this model is not comprehensive then it is incomplete.

Documents and experts usually play a part of the knowledge sources circumscribing the problem domain. There are several types of documents that may be used: legal documents, ones that describe business processes, databases of employees and customers, etc. Human experts may provide information on informal rules, conventions, relative importance of concepts, etc, in the given problem domain. Documents of listed types denote objects and formalize some relations in the problem domain concerned. To a first approximation they may be considered as local models of these relations.

The difficulty is that most local models are built using different approaches, because there is no unified approach that may be applied to a problem domain (excepting some narrow-ranged technical domains, where local models can be combined together into a global model using some strict mathematical rules; information systems built upon such problem domains are called “systems of automatic control”).

We are concerned here about information systems of a different kind – systems where the human element is of primary importance. Investigation into such kinds of problem domains is a type of empirical research, related to the “sciences of the artificial”.

Nowadays most CASE tools (Computer-Aided Software Engineering tools) can automatically build source program code for a projected information system, using some initial formal model of the problem domain (usually the model is represented as a framework, or graph). The urgent problem is to automate the process of building the formal model, e. g. to automate the process of cognition in the given problem domain.

Goals of the Research

The main purpose of this research is development of the special cognitive approach, referred to a class of Intellectual Modelling Technologies (IMT). This approach is designed for automating the process of information system development. Attention is focused on the very early stage of project development, the stage of analysis.

The problem domain of the class of IS under consideration includes a very large amount of legal documents (articles, assizes, bans, etc.), which regulate the status of objects, the behaviour of subjects related to an institution, etc. It also includes a settled system of document circulation. All this information, as a rule, is poorly structured. So, the development of a conceptual model of the problem domain (by means of UML language, for example) using knowledge from documents of these types, is a very difficult task and usually is done manually.

The suggested cognitive approach is aimed toward the problem of automating the conceptual-level model development by using loosely structured natural language documents.

Since the problem under consideration refers to a class of logical lexical systematization problems (as an example from the adjacent area of study we may take translation of natural language text into the language of predicate logic), it has no solution using only a computation system. That is why the suggested approach is developed to work in conjunction with the human user. Human interference is needed during the automated analysis of problem domain described in source documents. Nevertheless, some self-learning capabilities in the context of approach allow us to depend on the self-development of the analyzer during persistent dialogue with the human user, so that subsequently it could be able to solve similar tasks without direct human assistance.

The suggested approach is oriented to be utilized at the earliest stage of the information system project development process. As indicated on fig. 1, the suggested approach is supposed to be used at the very beginning of the spiral loop, at the boundary between the stages of requirements determination and analysis, also including the stage of analysis.

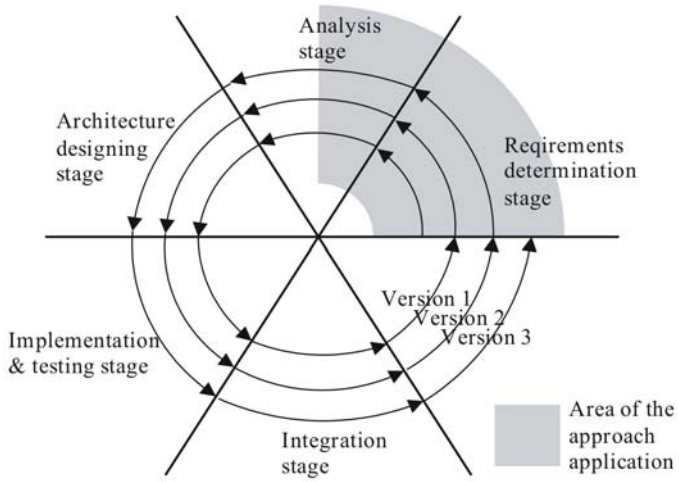

Figure 1. The Spiral Model of Software Life Cycle

It is important to mention that most existing IMT methods, used in CASE systems, automate, in general, stages of projecting, implementing, testing and integrating, but never touch the stages of requirements determination and analysis. Transition from the stage of requirements determination to the early stage of analysis is usually done by hand. Then the user develops a conceptual level model of the problem domain.

The model is developed usually using some special diagram language (UML language, for example). Conceptual level models describe a part of the real world for which the information system is being developed, so conceptual level class diagrams are right for describing the set of terms of the problem domain vocabulary.

When developing a model, the analyst usually processes by hand a large amount of documents referred to the problem domain in order to pick out key terms, properties, functions and relations between them. The proposed approach enables automation of this process. The intellectual cognitive analyzer being implemented according to the described approach should act as a user's assistant. It will do the most routine part of the work in the early stage of analysis.

The suggested approach also includes some other capabilities that let the computer become quite a good assistant for the human user not just at the beginning of analysis, but along its whole length. One of those capabilities, in particular, is the automatic problem domain thesaurus building during interaction with the user. And it is possible to use preinstalled thesauruses too, different ones for each problem domain, describing their specific components, features, etc.

Conceptual-level Modeling

As was said earlier, the purpose of the approach is automated construction of conceptual-level model diagrams of the problem domain. The UML language (static class section), was chosen as a model representing language, because it is the most popular standard for CASE tools nowadays.

UML static class diagrams define object classes and different kinds of static relations between them in the system. Also such things as class attributes, operations and relation limitations are usually shown on class diagrams.

There are three different points of view on how to interpret the sense of class diagrams:

- Conceptual-level point of view. If we take a class diagram from this point of view, then it reflects a set of terms and relations (called vocabulary) of the examined problem domain. Conceptual-level model considered independent from any software programming language.
- Specification-level point of view. In contrast to the above, this affects the software development range, but focuses attention over interfaces, not implementation. Looking at the class diagram from this point of view, designers have to do rather with types, not classes.

– Implementation-level point of view. In this case we really deal with graphical representation of the class structure of software. So the designer goes down to the level of implementation.

Understanding which point of view should be used and when, is extremely important either for developing or for reading class diagrams. Unfortunately, distinctions between them are not understood clearly, so the majority of developers often mix some different points of view when developing a diagram model.

The idea of the point of view on diagrams is not actually a formal part of UML language, but it is extremely important. UML constructions can be used with any of the three points of view in mind.

As has already been said, the suggested approach is going to be used for the automation of the process of conceptual-level problem domain model development. First of all, it is because of the fact that the approach should work at the most initial stage of IS development process. Apart from that, the nature of the documentation used in the problem domain of the considered range of IS (sphere of education) means that the description of objects and their mutual relations is of a sufficiently high level. This fact automatically determines the point of view on a problem domain as conceptual.

However, such a strict binding model to a conceptual level is not obligatory. In some cases the model can get an interpretation from some other point of view. This mainly depends on the nature of the source documents.

Conceptual-level diagrams describe the problem domain vocabulary. Of course, it is doubtful that diagrams developed using the suggested approach could be immediately used for generation of skeleton program code, but it can be used for subject domain database logic structure generation.

IES Architecture

Fig. 2 shows the diagram reflecting the principle according to which the projected system is organized.

Let us consider in more detail the principles assumed for the basis of the suggested approach.

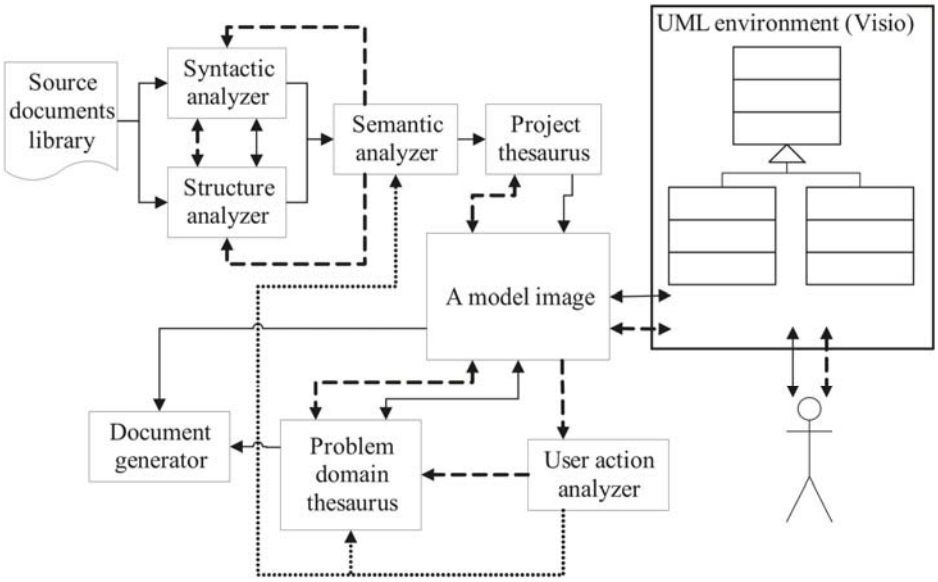

Figure 2. IES Architecture Framework

Natural language expresses relations between items in a problem domain in the form of statements. For example, the statement “children study at schools” binds together the concept “school” belonging to the class “educational institutions” and the concept “children” belonging to the class “person”. Any statement can be either correct, or wrong, when established during correlation with reality. So, statements singled out from source documents should be compared to the problem domain thesaurus which reflects the current actuality. In the case of detection of a discrepancy of the obtained propositions to ones from the problem domain thesaurus, the latter should be

brought into accord with reality, or the source proposition should be corrected in an appropriate way. When the system cannot make such a decision independently, it can apply for the human user's assistance.

The proposition (statement) is an expression that claims or disclaims the existence of an item, the relation between an item and its attribute, or the relation between two items. A sentence is the language form of the proposition. Propositions in natural language texts are expressed by narrative sentences, for example: “institutions of primary vocational training may be state, municipal and private”. The proposition of connexion of an item and its attribute consists of propositional subject, and a predicate reflecting an attribute of an item. Except for subject and predicate, the proposition includes a copula which can be put into words (for example, “not is”, “is”, etc.).

Depending on what is claimed or disclaimed in the proposition – either the existence of an item, or the relation between an item and its attribute, or the relation between two items – it may be classified as attributive proposition, proposition with relation and existential proposition. A proposition is called compound if it consists of several simple propositions, combined together in an expression.

A conceptual-level model is usually developed using source natural language description. Sentences in this description are propositions of listed types. Some of them concern certain objects; others are general as they concern some class of objects in the problem domain. Source documents consist of compound sentences that describe objects and relations between them in the problem domain.

In the course of linguistic analysis using knowledge of language structure, initial compound sentences are split into simple propositions of three listed types, and the type of each proposition can be determined during the process of decomposition.

The whole totality of concepts and relations between them, expressed by means of natural language, forms a system thesaurus. Thus, we can say it schematically represents the matter of the source documents text. The idea of a thesaurus is frequently applied to problems of semantic search in documents. Within the suggested approach, another variant of its application is offered.

Concepts are extracted from source documents during the linguistic analysis process. One of the basic relation types that make a thesaurus hierarchical, is the relation type named “is a”. It realizes the idea of generalization, allowing reference of a narrower concept to a wider one.

Another relation type named “referred to” designates a certain reference between terms. It can be marked by a verb (predicate) extracted from the source sentence (this verb should describe the nature of the relation with certainty). This mark can also be a word-combination, consisting of a verb and a noun: “is operation”, “is an attribute”.

To avoid the possibility of the appearance of a vicious circle of interdependence of terms in a thesaurus, there are some rules to be obeyed:

- no term can be wider than itself, either directly, or indirectly (this limits the usage of the “is a”);
- no term can be “referred to” a term which is wider or narrower than itself, either directly, or indirectly.

The structure of the thesaurus can be represented by a graph (semantic net), its nodes correspond to terms, and arches are relations between terms. One set of arches forms a directed acyclic graph for the relation of generalization (“is a”). Another set forming the directed graph, represents the relation of referred terms (“referred to”). Relation types “is a” and “referred to” form subgraphs.

Principles of IES Operation

The thesaurus of the model should be populated and refreshed using an automatic mode. Thus there are the certain difficulties concerning natural language text processing. To overcome these difficulties successfully, the approach offers the multilevel circuit of text processing using the relaxation method to eliminate ambiguities.

At the initial stage of text processing the syntactic analyzer (figure 2) works. It implements the syntactic analysis and decomposition of compound sentences to the simple propositions consisting of subject, predicate and object. While these operations are being accomplished, the semantics of the sentence is not taken into account. During decomposition, the text of the source documents is transformed into a set of simple statements (propositions) of

three listed types (attributive proposition, proposition with relation, existential proposition) which then can be easily subjected to semantic analysis.

It is important to note that relations between concepts are not necessarily conveyed syntactically in text. They can also be conveyed by the structure of the document. There are two types of structural compositions most frequently used in documents: table structure, determines attributive relations; list structure, determines relations of various kinds, between the concept located in the list heading and concepts located in lines.

In order to assure the completeness of analysis it is necessary to allocate relations, set by structures of listed types. This task is done by the structural analyzer, whose output, as well as for syntactic analyzer, consists of simple propositions reflecting relations between concepts. The analyzer generates them using structural information extracted from the source text as the basis.

The semantic analyzer obtains the data processed by syntactic and structural analyzers, handles them for its turn and populates the system thesaurus with prepared data. If the semantic analyzer finds any variance in source data, caused by its ambiguity or uncertainty, it can address previous level of processing – syntactic or structural analyzer – with the requirement to give another variant of the text treatment. This idea accords with principles of the relaxation method. Some missing branches of concept relations may also be evoked from the existing thesaurus knowledge base.

There is one more task assigned to the semantic analyzer – to eliminate insignificant data. In fact the final model should not be formed by the whole totality of concepts and relations, allocated in the initial documents. First of all, some concepts may just slightly touch the scope of the given problem domain. Sometimes some errors in allocation of concepts and relations may take place because of text specificity or its author's verbiage. Anyway, some mechanism is required that could free the user from dealing with a lot of insignificant details. To achieve this, the semantic analyzer uses a special self-learning filter as a part of the project thesaurus. This filter determines a list of concepts that should not be included in the thesaurus. Relations of a special type “not relevant” may also be settled between the concepts in the thesaurus in order to solve the problem more effectively.

The filter is trained by tracking actions which are user made when editing a diagram. This way we can reveal insignificant concepts and relations in the problem domain to use this knowledge later.

We need to mention that there is one more important opportunity the approach can offer: an opportunity of distribution “on a turn-key basis” of an IS designing tool assigned for usage in the context of a certain problem domain. Such a tool would possess a prepared thesaurus establishing the set of basic concepts and relations and include a trained semantic filter focused over the scope of the problem domain being aimed at. In the architecture framework of IES which is being developed according to the suggested approach, this thesaurus is represented by the separate component called “Problem domain thesaurus” (figure 2).

The project thesaurus directly delivers data needed for production of model diagrams. The structure and sense of the thesaurus content allows translation of it into the model diagram. This is in spite of the fact that there are some minor distinctions between specifications of diagrams that could be used within the approach: UML diagrams and ER diagrams.

Diagrams are displayed in some external modelling environment which is connected to IES through the special buffer module of the model image. Of course, the user may want to correct the obtained model diagram, which is initially generated by the system. But nevertheless, it continues to cooperate with the user, helping him to perform the work.

Upon the user's demand it can generate some new fragments of the model diagram, if there are any new source documents obtained, or expand the model diagram in order to restore missing relations, applying knowledge from the problem domain and the project thesauruses, etc.

The system also traces user's actions made during model diagram editing. Such a feedback mechanism is absolutely necessary for implementing the idea of self-training as applied to the problem domain thesaurus and

the semantic filter. Actually, during editing of the model diagram, the user "teaches" the system, providing it with the information about concepts and relations that are of first interest to him and ones that should be excluded from consideration. In such a way, the problem domain thesaurus containing the most authentic and clean information on key concepts and typical relations between them is built. It is populated automatically during editing of the model diagram. Thus, the resulting model diagram and successive modifications made by the user are also a source of information for the IES.

The system tries to recognize semantics in the model diagrams. So a diagram which the user works with is not a senseless set of blocks and connections for a computer any more. Attention is focused on names of elements, their structure, interfacing, etc. All these aspects are analyzed by the system.

Objects and relations allocated in a problem domain, organize a model. When the diagram is built, they remain connected with texts in the source documents library. It is necessary for the user to have an opportunity to supervise the correctness of the constructed model, verifying it directly with the key information from source documents. Reverse referencing from source documents to elements of a model diagram is also needed, because documents are not something immutable. The documents library has a dynamic nature – precepts may be cancelled, or changed in some points, etc. Direct and reverse referencing between source texts and the model assure an opportunity of efficient model updating.

Examples

Now we give an example demonstrating some aspects of the approach.

Please note that the approach is being developed for use jointly with the Russian language, where the concepts' mutual interdependence in sentences is expressed much less ambiguously than in English, at the syntax level.

Let us show how a certain expression is going to be analyzed by the system:

"Educational institutions with government accreditation grant certificates to persons who passed attestation".

During syntactic analysis the given sentence is split into some connected simple statements which can be easily represented by the semantic network shown on fig. 3.

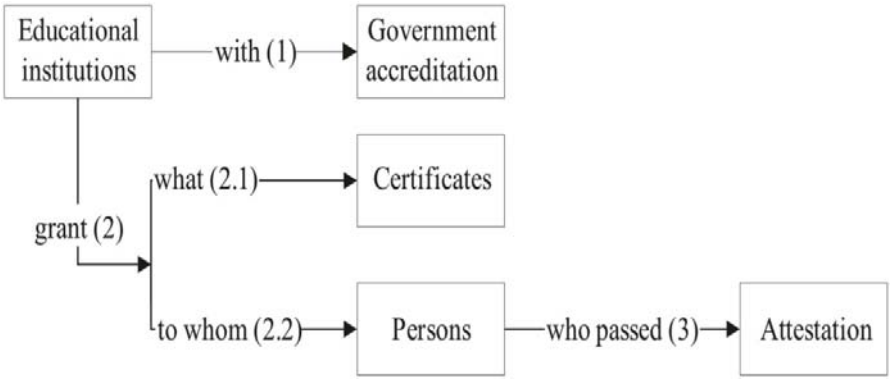

Figure 3. Semantic Network Representing Sample Sentence Structure

The semantic analyzer qualifies propositions (1, 2 and 3) such as ones with relations. Thus the verb predicate representing the action "grant" is interpreted by the semantic filter as an operation (class method). But let us assume that such interpretation is not known to semantic filter.

Simple propositions obtained which form marked section of a semantic network after the stage of semantic analysis, are directed to the problem domain thesaurus.

Propositions with relations of such a kind are displayed in the model as objects connected by the relation "referred to"; connection is directed from a subject to an object and represents the predicate (see fig. 4).

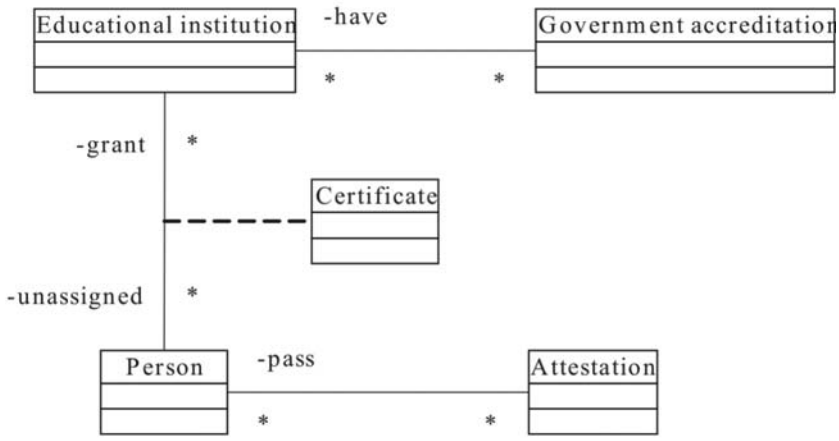

Figure 4. Model Framework Created on the Sentence

Part of the model received as a result of analysis of a given sentence, could be automatically attached to the existing model by a set of "is a" connections, revealed by the semantics comparison.

Besides that, if the problem domain thesaurus contains information about other connections between these objects and ones in the problem domain, these connections will also be restored in the model.

So, let us return to the necessity that the action "grant" should be interpreted as a method.

If it does not happen automatically, then the user manually creates the method "grant" in the object "Education institution". After that, as a result of the semantics comparison of the operation name assigned by the user with the text of source sentence, the semantic filter is trained to interpret the verb "to grant" as the method (operation) at a later time.

Analyzing a similar text subsequently, the system should automatically add a corresponding object operation to the model. The thesaurus of the model is populated and refreshed in an automatic mode.

Bibliography

[Aiello, 2000] M. Aiello, C. Monz, L. Todoran. Combining Linguistic and Spatial Information for Document Analysis, In: Content-Based Multimedia Information Access. CID, 2000, pp. 266-275.

[Connolly, 1999] T.M. Connolly, C.E. Begg. Database Systems. A Practical Approach to Design, Implementation, and Management. Addison Wesley Longman, Inc, 1999.

[Fowler, 1997] M. Fowler, C. Scott. UML Distilled: Applying the Standard Object Modeling Language. Addison Wesley Longman, Inc, 1997.

[Johnson-Laird, 1983] P.N. Johnson-Laird. Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness. Harvard University Press, 1983.

[Katz , 1964] J.J. Katz, J.A. Fodor. The Structure of Language, Prentice-Hall, 1964.

[Larman, 2000] C. Larman. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design. Prentice Hall, Inc, 2000.

[Miller, 1990] G. Miller. Wordnet: An on-line lexical database. In: International Journal of Lexicography, 3(4), 1990.

Authors' Information

Maxim Loginov - Perm State University, Assistant of the Department of Computer Science; PSU, 15, Bukirev St., Perm, 614990, Russia; e-mail: login@perm.ru

Alexander Mikov – Institute of Computing, Director; 40/1-28, Aksaiskaya St., Krasnodar, Russia; e-mail: alexander_mikov@mail.ru